

Jan M. Rabaey
Anantha Chandrakasan
Borivoje Nikolic

© Digital Integrated Circuits^{2nd}

1

Devices

CSCE 613 – Week 12 Fall 2005

Introduction to CMOS VLSI Design

CMOS Analysis Methods

Adapted/extended by James P. Davis, Ph.D.
Dept. of Computer Science & Engineering
University of South Carolina

Topics of Week 12

- **Chapter 6 text material – Analysis method.**
 - We look more closely at Logical Effort as a means to carry out the manual analysis of the gate and switch-level structures prior to laying out the individual cells.
 - Note that Logical effort is a means for “refining” the analysis you may have already done (using the more basic methods we’ve discussed up to this point).
 - The focus here is to be able to perform this analysis on your cells, from which you would then carry out the analysis at the next level up in your design hierarchy—the circuit level. Here, we are interested in the delay analysis, transistor sizing, buffer placement, etc., given that we are going to use the TSMC 0.35 micron library from Mentor Graphics.

- **Insert E Material (pp. 428-434)**
 - Characterizing logic cells (another take on the analysis).
 - Using a design library (now that we have the ADK library supplied by Mentor, we can discuss this in terms of the TSMC 0.35 micron library we’ll be using).

Complementary CMOS Analysis - Speed

- Frequently, input capacitance of a logic path is constrained (effect of fan-in).
- Logic also has to drive some capacitance (effect of fan-out).
- How do we size the arithmetic datapath in our projects to achieve maximum speed?
- We have already solved this for the inverter chain – can we generalize it for any type of logic? Yes, and here's how:

Logical Effort

- Inverter has the smallest logical effort and its intrinsic delay is composed of all static CMOS gates.
- *Logical effort* of a gate presents the ratio of its input capacitance to the inverter capacitance when sized to deliver the same current.
 - We use the inverter as the gauge for sizing the NAND and NOR (m-inputs) structures used in our arithmetic circuits.
- Logical effort increases with the gate complexity.
 - This is defined in terms of number of inputs (i.e., fan-in), fan-out.

Composite Analysis - Delay in a Logic Gate

Gate delay:

$$d = h + p$$

effort delay intrinsic delay

Effort delay:

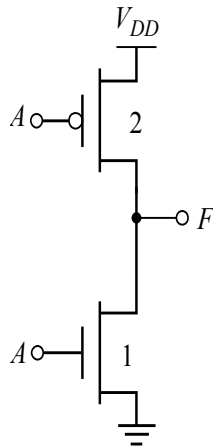
$$h = g f$$

logical effort effective fanout = C_{out}/C_{in}

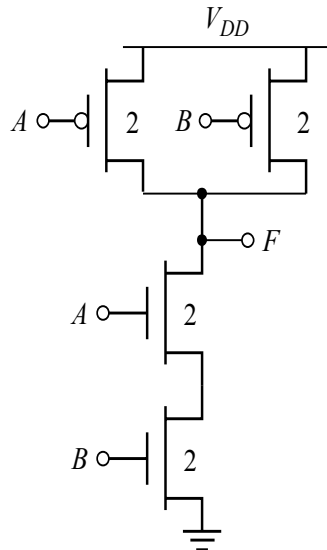
Logical effort is a function of topology, independent of sizing.
Effective fanout (electrical effort) is a function of load/gate size.

Logical Effort

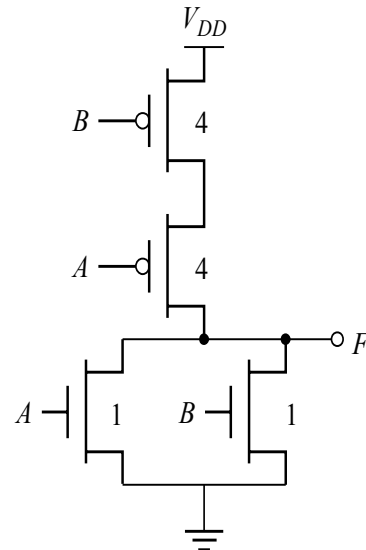
Logical effort is the ratio of input capacitance of a gate to the input capacitance of an inverter with the same output current



Inverter
 $g = 1$



2-input NAND
 $g = 4/3$



2-input NOR
 $g = 5/3$

Definition of Logical Effort

$$\begin{aligned} \text{Delay} &= k \cdot R_{\text{unit}} C_{\text{unit}} \left(1 + \frac{C_L}{\gamma C_{\text{in}}} \right) \\ &= \tau(p + g \cdot f) \end{aligned}$$

p – intrinsic delay ($3k R_{\text{unit}} C_{\text{unit}} \gamma$) – gate parameter $\neq f(W)$
 g – logical effort ($k R_{\text{unit}} C_{\text{unit}}$) – gate parameter $\neq f(W)$
 f – effective fanout

Normalize everything to an inverter:

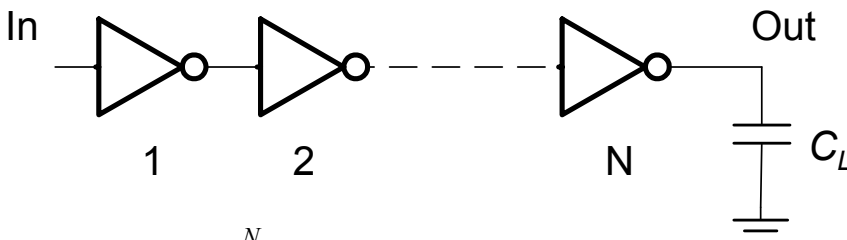
$$g_{\text{inv}} = 1, p_{\text{inv}} = 1$$

Divide everything by τ_{inv}

(everything is measured in unit delays τ_{inv})

Assume $\gamma = 1$.

Buffer Example



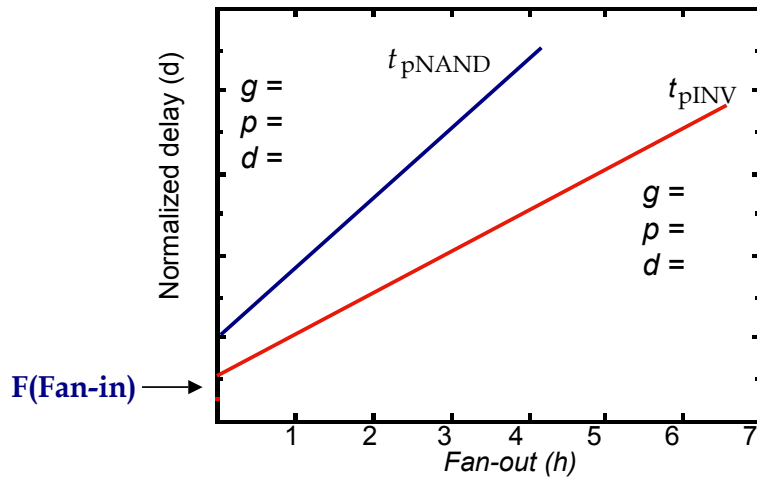
$$\text{Delay} = \sum_{i=1}^N (p_i + g_i \cdot f_i) \quad (\text{in units of } \tau_{\text{inv}})$$

For given N : $C_{i+1}/C_i = C_i/C_{i-1}$

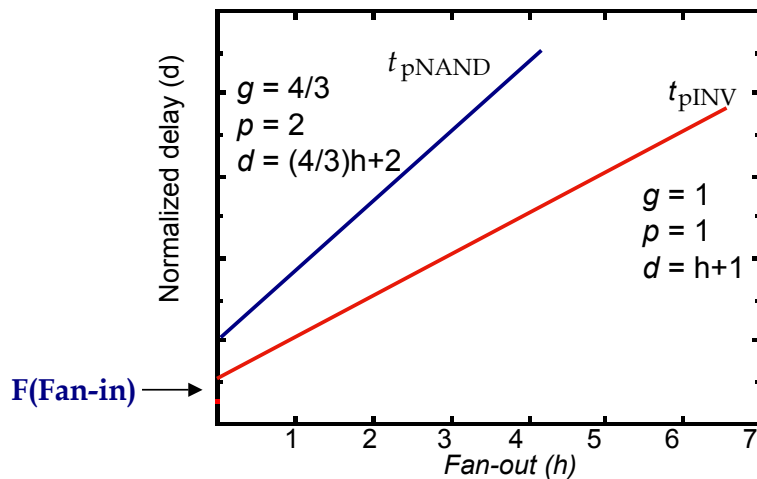
To find N : $C_{i+1}/C_i \sim 4$

How to generalize this to any logic path?

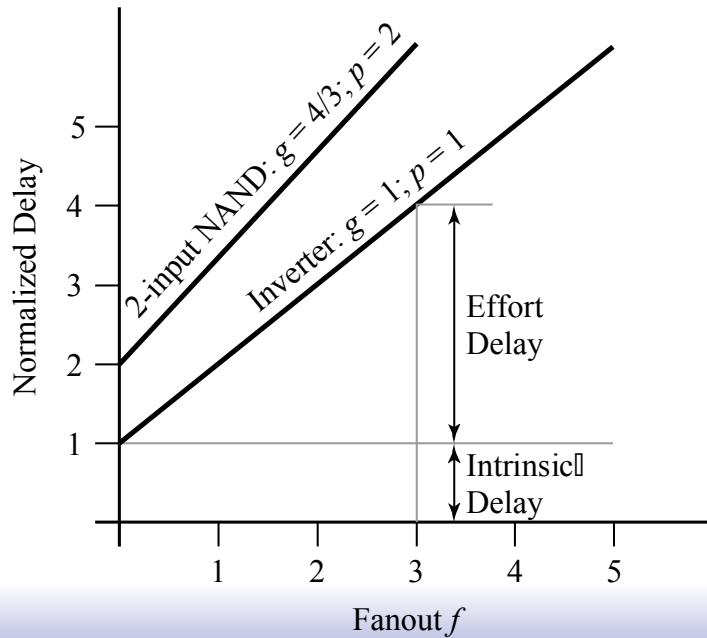
Logical Effort of Gates



Logical Effort of Gates



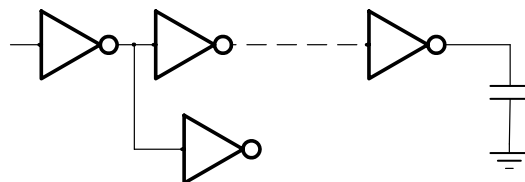
Logical Effort of Gates



Add Branching Effort

Branching effort:

$$b = \frac{C_{on-path} + C_{off-path}}{C_{on-path}}$$



Multistage Networks

$$Delay = \sum_{i=1}^N (p_i + g_i \cdot f_i)$$

Stage effort: $h_i = g_i f_i$

Path electrical effort: $F = C_{out}/C_{in}$

Path logical effort: $G = g_1 g_2 \dots g_N$

Branching effort: $B = b_1 b_2 \dots b_N$

Path effort: $H = GFB$

Path delay $D = \sum d_i = \sum p_i + \sum h_i$

Optimum Effort per Stage

When each stage bears the same effort:

$$h^N = H$$

$$h = \sqrt[N]{H}$$

Stage efforts: $g_1 f_1 = g_2 f_2 = \dots = g_N f_N$

Effective fanout of each stage: $f_i = h/g_i$

Minimum path delay

$$\hat{D} = \sum (g_i f_i + p_i) = NH^{1/N} + P$$

Optimal Number of Stages

For a given load,
and given input capacitance of the first gate
Find optimal number of stages and optimal sizing

$$D = NH^{1/N} + Np_{inv}$$

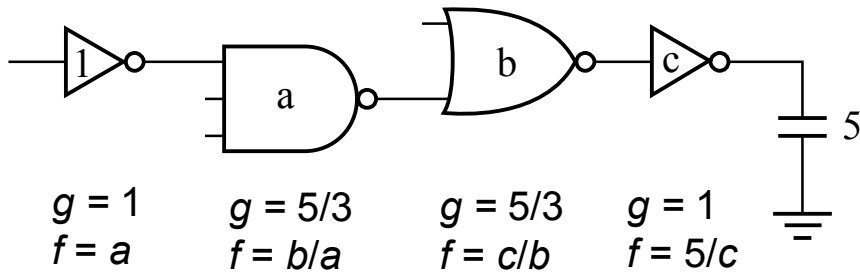
$$\frac{\partial D}{\partial N} = -H^{1/N} \ln(H^{1/N}) + H^{1/N} + p_{inv} = 0$$

Substitute 'best stage effort' $h = H^{1/\hat{N}}$

Logical Effort

Gate Type	Number of Inputs			
	1	2	3	n
Inverter	1			
NAND		4/3	5/3	(n + 2)/3
NOR		5/3	7/3	(2n + 1)/3
Multiplexer		2	2	2
XOR		4	12	

Example: Optimize Path



Effective fanout, $F =$

$G =$

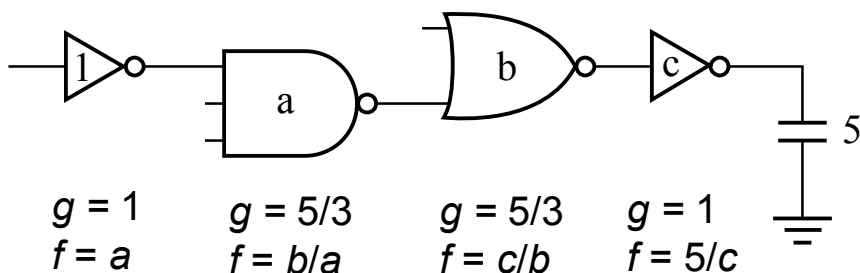
$H =$

$h =$

$a =$

$b =$

Example: Optimize Path



Effective fanout, $F = 5$

$G = 25/9$

$H = 125/9 = 13.9$

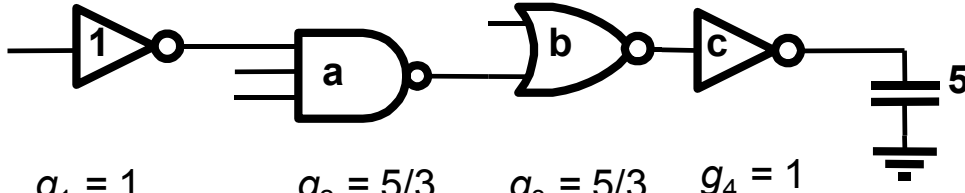
$h = 1.93$

$a = 1.93$

$b = ha/g_2 = 2.23$

$c = hb/g_3 = 5g_4/f = 2.59$

Example: Optimize Path



$$g_1 = 1$$

$$g_2 = 5/3$$

$$g_3 = 5/3$$

$$g_4 = 1$$

Effective fanout, $H = 5$

$$G = 25/9$$

$$F = 125/9 = 13.9$$

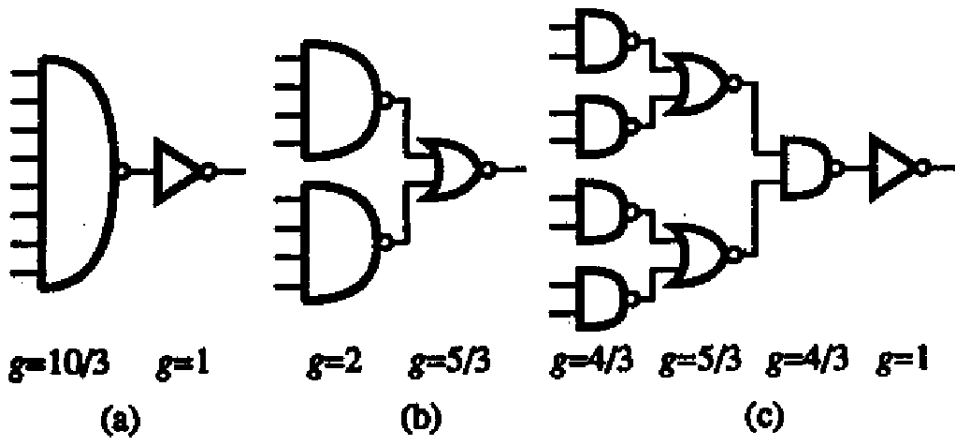
$$f = 1.93$$

$$a = 1.93$$

$$b = fa/g_2 = 2.23$$

$$c = fb/g_3 = 5g_4/f = 2.59$$

Example – 8-input AND



Method of Logical Effort

- Compute the path effort: $F = GBH$
- Find the best number of stages $N \sim \log_4 F$
- Compute the stage effort $f = F^{1/N}$
- Sketch the path with this number of stages
- Work either from either end, find sizes:

$$C_{in} = C_{out} * g/f$$

Reference: Sutherland, Sproull, Harris, "Logical Effort, Morgan-Kaufmann 1999.

Week 12 Summary

Key Definitions of Logical Effort

Term	Stage expression	Path expression
Logical effort	g (see Table 1)	$G = \prod g_i$
Electrical effort	$h = \frac{C_{out}}{C_{in}}$	$H = \frac{C_{out (path)}}{C_{in (path)}}$
Branching effort	n/a	$B = \prod b_i$
Effort	$f = gh$	$F = GBH$
Effort delay	f	$D_F = \sum f_i$
Number of stages	1	N
Parasitic delay	p (see Table 2)	$P = \sum p_i$
Delay	$d = f + p$	$D = D_F + P$

From: Sutherland, Sproull & Harris, *Logical Effort: Designing Fast CMOS Circuits*, Morgan Kaufmann, 1999.